

IN THE CLAIMS:

Please amend the claims as follows.

1. (Currently Amended) A method for implementing a functional memory, ~~in which~~wherein memory data is stored as data units for each of which a dedicated storage space is assigned in the memory, ~~in accordance with which~~the method comprising:

~~—implementing the a memory is implemented as~~ a directory structure comprising a tree-shaped hierarchy having nodes at several different levels, wherein an individual node ~~can be~~includes (i) a trie node associated with a logical table wherein an individual element ~~may contain~~contains a pointer pointing to a lower node in the tree-shaped hierarchy ~~and wherein an individual element may also be empty, in which case the content of the element corresponds to a nil pointer, the a number of elements in the logical table corresponding to a power of two, or (ii) a bucket containing at least one element in such a way that the~~wherein a type of an said individual element in the bucket is selected from a group including a data unit, a pointer to a stored data unit, a pointer to another directory structure and said another directory structure;

performing address computation ~~performed in the directory structure comprises the steps of by~~

(a) selecting in the node at ~~the an~~an uppermost level of the tree-shaped hierarchy a given number of bits from ~~the a~~a bit string formed by ~~the employed search keys employed, forming from the selected given number of bits a search word with which the~~for seeking an address of ~~the a~~a next node is sought in the individual node,

and proceeding to said next node,

(b) selecting a predetermined number of bits from the unselected bits in the bit string formed by the employed search keys ~~employed~~ and forming from the ~~selected~~ predetermined number of bits another search word with which the for seeking another address of a further new node at a lower level is ~~sought~~ from the logical table of the individual node that has been accessed,

repeating step (b) until an element containing a nil pointer is encountered or until the address of the new node at a said lower level is the address of a said bucket,

wherein the nodes to which a given node contains pointers are child nodes of said given node and the nodes to which the child nodes contain pointers are grandchild nodes of said given node; and

~~characterized by~~

implementing trie nodes as quad nodes of four elements, and replacing in at least part of the directory structure groups of successive nodes by compressed nodes ~~in~~ such a way that by

(a) replacing an individual group comprising a given quad node and its child nodes ~~is replaced~~ by a node whose logical table has 16 elements, and

(b) forming a compressed node known per se ~~is formed~~ from said node of 16 elements by physically storing in the compressed node ~~only~~ non-nil pointers and ~~in addition~~ a bit pattern on the a basis of which the a physical storage location in the compressed node, corresponding to the search word, ~~can be~~ is determined.

2. (Currently Amended) A method as claimed in claim 1, ~~characterized in that~~further comprising performing replacement is carried out in the directory structure on all groups in which the quad node has two child nodes.

3. (Currently Amended) A method as claimed in claim 1, ~~characterized in that~~further comprising performing replacement is carried out in the directory structure on all groups in which the quad node has up to eight grandchild nodes ~~at most~~.

4. (Currently Amended) A method as claimed in claim 2, ~~characterized in that~~further comprising setting an upper limit is set for the number of pointers in the compressed node, wherein when said limit is exceeded, the compressed node is ~~again decompressed to a~~another quad node and child nodes.

5. (Currently Amended) A method as claimed in claim 4, ~~characterized in that~~further comprising employing eight pointers is employed as said upper limit.

6. (Currently Amended) A method as claimed in claim ~~14~~4, ~~characterized in that~~further comprising employing ten pointers is employed as said upper limit.

7. (Currently Amended) A method as claimed in claim 2, ~~characterized~~
~~in that~~ further comprising performing compression ~~is additionally carried out on~~ at least
some of the quad nodes (N80...N82) in the structure ~~in such a way so that~~ only non-nil
pointers are physically stored in the node and ~~in addition a bit pattern (BP2) on the a~~
basis of which the physical storage location in the node, corresponding to the search
word, ~~can be~~ is determined.

8. (Currently Amended) A method as claimed in claim 1, ~~characterized~~
~~in that~~ further comprising storing the non-nil pointers ~~are stored in the compressed node~~
~~in succession in the a~~ same order that ~~they~~ the non-nil pointers have in said logical table.

9. (Currently Amended) A method as claimed in claim 8,
~~characterized in that~~ wherein the bit pattern has one bit for each element in
the table, ~~each said one~~ bit indicating whether the corresponding element contains a
nil pointer or a non-nil pointer.

10. (Currently Amended) A method as claimed in claim 8, ~~characterize~~
~~d in that~~ further comprising reserving space ~~is reserved for the bit pattern in all said~~
trie nodes of the directory structure.

11. (Currently Amended) A method as claimed in claim 8, ~~characterized~~
~~in that~~ further comprising reserving space is reserved for the bit pattern in the compressed
nodes ~~only~~.

12. (Currently Amended) A method for implementing a functional memory, ~~in~~
~~which~~ wherein memory data is stored as data units for each of which a dedicated storage
space is assigned in the memory, ~~in accordance with which~~ the method comprising:

implementing the memory is implemented as a directory structure
comprising a tree-shaped hierarchy having nodes at several different hierarchy levels,
wherein an individual node ~~can be~~ includes (i) an internal node associated with a logical
table wherein an individual element ~~may contain~~ contains a pointer pointing to a lower
node in the tree-shaped hierarchy and ~~wherein an individual element may also be empty,~~
~~in which case the content of the node corresponds to a nil pointer,~~ the a number of
elements in the logical table corresponding to a power of two, or (ii) a leaf containing an
element ~~the of a type of which is selected from a group including a pointer to a stored~~
data unit, a data unit, and a pointer to a node in another directory structure;

performing address computation performed in the directory structure
~~comprises the steps of by~~

(a) selecting in the individual node at the an uppermost level of the
tree-shaped hierarchy a given number of bits from ~~the a~~ bit string formed by ~~the~~
employed search keys employed, forming from the selected given number of bits a search

word ~~with which the~~ to seek an address of the ~~a~~ next node ~~is sought in the~~ individual node, and proceeding to said next node,

(b) ~~selecting a~~ another given number of bits from ~~the~~ unselected bits in the bit string formed by the employed search keys ~~employed~~, and forming from the ~~selected~~ another given number of bits ~~a~~ another search word ~~with which the~~ to seek another address of a further new node at a lower level ~~is sought from the~~ logical table of the individual node that has been accessed, and

repeating step (b) until an empty element is encountered or until the address of the new node at a lower level is the address of ~~a~~ the leaf,

wherein the nodes to which a given node ~~contains~~ includes pointers are child nodes of said given node and the nodes to which the child nodes contain pointers are grandchild nodes of said given node; and

~~characterized by~~

implementing internal nodes as quad nodes having four elements, and replacing, in at least part of the directory structure, groups of successive nodes by compressed nodes ~~in such a way that~~ by

replacing an individual group comprising a given quad node and its child nodes ~~is replaced by~~ a node whose node logical table has 16 elements, and

forming a compressed node known per se ~~is formed from~~ said node of 16 elements by physically storing in the compressed node ~~only~~ non-nil pointers and ~~in addition~~ a bit pattern on ~~the~~ a basis of which ~~the~~ a physical storage location in the

compressed node, corresponding to the search word, ~~can be~~is determined.

13. (Currently Amended) A method as claimed in claim 12, ~~characterized in that~~ further comprising performing replacement ~~is carried out~~ in the directory structure on all groups in which the quad node has two child nodes.

14. (Currently Amended) A method as claimed in claim 12, ~~characterized in that~~ further comprising performing replacement ~~is carried out~~ in the directory structure on all groups in which the quad node has up to eight grandchild nodes ~~at most~~.

15. (Currently Amended) A method as claimed in claim 13, ~~characterized in that~~ further comprising setting an upper limit ~~is set for the~~ a number of pointers in the compressed node, wherein

when said upper limit is exceeded, the compressed node is ~~again~~ decompressed to a quad node and child nodes.

16. (Currently Amended) A method as claimed in claim 15, ~~characterized in that~~ further comprising employing eight pointers ~~is employed as said~~ upper limit.

17. (Currently Amended) A method as claimed in claim 15, ~~characterized in that~~ further comprising employing ten pointers is ~~employed as said upper limit.~~

18. (Currently Amended) A method as claimed in claim 13, ~~characterized in that~~ further comprising performing compression is additionally ~~carried out on~~ at least some of the quad nodes in the structure ~~in such a way so that only~~ the non-nil pointers are physically stored in the compressed node and in addition a bit ~~pattern (BP2) on the a~~ basis of which the physical storage location in the node, ~~corresponding to the search word, can be~~ is determined.

19. (Currently Amended) A method as claimed in claim 12, ~~characterized in that~~ further comprising storing the non-nil pointers are stored ~~in the compressed node in succession in the a~~ same order that they have as in said node logical table.

20. (Currently Amended) A method as claimed in claim 19, ~~characteri~~ ~~zed in that~~ wherein the bit pattern has one bit for each element in the node logical table, each bit indicating whether the each ~~corresponding~~ element contains a nil pointer or a non-nil pointer.

21. (Currently Amended) A method as claimed in claim 19, ~~characterized in that~~ further comprising reserving space is reserved for the bit pattern in all trie nodes of the directory structure.

22. (Currently Amended) A method as claimed in claim 19, ~~characterized in that~~ further comprising reserving space is reserved for the bit pattern in the compressed nodes ~~only~~.

23. (Currently Amended) A memory arrangement for storing data units, said memory arrangement comprising:

a directory structure ~~in which~~ wherein progress is made by using search words formed from a bit string constituted by ~~the~~ search keys employed in each case, said directory structure comprising a tree-shaped hierarchy having nodes at several different hierarchy levels,

wherein an individual node ~~can be~~ is (i) a trie node associated with a logical table wherein an individual element ~~may contain~~ contains a pointer pointing to a lower node in the tree-shaped hierarchy ~~and wherein an individual element may also be empty, in which case the content of the element corresponds to a nil pointer, the~~ a number of elements in the logical table corresponding to a power of two, or (ii) a bucket containing at least one element ~~in such a way~~ so that the a type of an individual element in the bucket

is selected from a group including a data unit, a pointer to a stored data unit, a pointer to a node in another directory structure, and said another directory structure,

~~characterized in that~~

wherein some of the trie nodes are quad nodes ~~whose~~ having a node logical table ~~has that includes~~ four elements and some ~~are of the trie nodes whose are nodes~~ having a node logical table ~~has that includes~~ 16 elements and ~~in which only~~ wherein non-nil pointers are physically stored in addition to a bit pattern (BP1) on the a basis of which the a physical storage location in the node, corresponding to the search word, ~~can be~~ is determined.

24. (Currently Amended) A ~~method~~ memory arrangement as claimed in claim 23, ~~characterized in that~~ wherein at least some of said quad nodes store physically ~~only~~ those pointers that are non-nil pointers and ~~in addition~~ a bit pattern (BP2) on the basis of which the physical storage location in the node, corresponding to the search word, ~~can be~~ is determined.

25. (Currently Amended) A memory arrangement for storing data units, said memory arrangement comprising:

a directory structure ~~in which~~ wherein progress is made by using search words formed from a bit string constituted by the search keys employed in each case, said directory structure comprising a tree-shaped hierarchy having nodes at several different

hierarchy levels, wherein an individual node ~~can be~~is (i) an internal node associated with a logical table wherein an individual element ~~may contain~~contains a pointer pointing to a lower node in the tree-shaped hierarchy ~~and wherein an individual element may also be empty, in which case the content of the element corresponds to a nil pointer,~~ a the number of elements in the logical table corresponding to a power of two, or (ii) a leaf containing at least one element of a type selected from a group including a pointer to a stored data unit and a pointer to a node in another directory structure,

~~e-h-a-r-a-c-t-e-r-i-z-e-d-i-n-t-h-a-t~~

wherein some of the trie nodes are quad nodes ~~whose~~ having a node logical table ~~has~~ including four elements and some of the trie nodes are nodes ~~whose~~ logical ~~having a node logical table has~~ that includes 16 elements and ~~in which~~ only wherein non-nil pointers are physically stored in addition to a bit pattern (BPI) ~~on the~~ a basis of which ~~the~~ a physical storage location in the node, corresponding to the search word, ~~can be~~is determined.

26. (Currently Amended) A ~~method~~ memory arrangement as claimed in claim ~~23~~25, ~~e-h-a-r-a-c-t-e-r-i-z-e-d-i-n-t-h-a-t~~ wherein at least some of said quad nodes store physically ~~only~~ those pointers that are non-nil pointers and ~~in addition~~ a bit pattern (BP2) on the basis of which the physical storage location in the node, corresponding to the search word, ~~can be~~is determined.

Please add new claims 27-30, as follows:

27. (New) A method as claimed in claim 1, further comprising determining said individual element is empty, wherein a content of the individual element corresponds to said nil pointer.

28. (New) A method as claimed in claim 12, further comprising determining said individual element is empty, wherein a content of the individual node corresponds to said nil pointer.

29. (New) A memory arrangement as claimed in claim 23, wherein said individual element is empty and a content of said individual element corresponds to a nil pointer.

30. (New) A memory arrangement as claimed in claim 25, wherein said individual element is empty and a content of said individual element corresponds to nil pointer.